Rwordseg使用说明

Jian Li*

1 简介及安装

1.1 简介

Rwordseg 是一个R环境下的中文分词工具，使用rJava调用Java分词工具Ansj。

Ansj 也是一个开源的Java 中文分词工具，基于中科院的ictclas 中文分词算法，采用隐马尔科夫模型（Hidden Markov Model, HMM）。作者孙健重写了一个Java版本，并且全部开源，使得Ansi 可用于人名识别、地名识别、组织机构名识别、多级词性标注、关键词提取、指纹提取等领域，支持行业词典、用户自定义词典。

详细信息可以参考作者孙健的专访（http://blog.csdn.net/blogdevteam/article/details/8148451）以及项目的Github 地址（https://github.com/ansjsun/ansj_seg）。

当前版本的Rwordseg包完全引用了Ansj 包，只是简单提供了R的接口，并根据R中处理文本的习惯进行了调整，在此对原作者孙健表示强烈的敬意！之所以没有命名为ransj，是因为以后可能还会纳入其他的分词工具或者自己开发一些新的功能。

1.2 安装

当前版本在Rforge（https://r-forge.r-project.org/R/?group_id=1054）进行维护，可以直接安装最新版。如果是老版的R，使用“source”的方式进行安装。

```
> install.packages("rJava")
> install.packages("Rwordseg", repos="http://R-Forge.R-project.org")
> #install.packages("Rwordseg", repos="http://R-Forge.R-project.org", type="source")
```

该包依赖于rJava包和Java环境，在安装之前需要确保JRE已经安装，并且正确地设置了环境变量。当前版本的R包在JRE 1.6.0_32下测试通过。如果不确定本机是否有JRE，可以搜索Oracle网站的JRE进行安装。

*电子邮件：rweibo@sina.com

以Windows系统为例，假设 JRE 的安装目录为 D:\jdk1.6.0_32\jre，R的安装目录为 D:\R\R-2.15.2。需要将以下路径添加到 PATH 环境变量中：

- D:\jdk1.6.0_32\jre\bin
- D:\jdk1.6.0_32\jre\bin\server
- D:\jdk1.6.0_32\jre\bin\client
- D:\R\R-2.15.2\library\rJava\jri

注意查看 JRE 的安装目录中 jvm.dll 是位于 server 还是 client 子文件，确保该文件夹位于 PATH 环境变量即可。全部设置完毕后可以正常导入R包：

```
> require(Rwordseg)
```

```
Loading required package: Rwordseg
Loading required package: rJava
# Version: 0.0-4
```

2 与其他分词包的对比

目前开源的分词工具有很多，但是集成到R中的不多。在这里我们对目前可用的工具进行简单的介绍，并针对一个例子比较分词的结果：

```
> teststring1 <- "小都督促织女将R语言学习到底！"
> teststring2 <- "小都是统计之都的昵称:)"
```

2.1 mmseg4j

目前最常用的R分词包可能是基于mmseg4j 的rmmseg4j ，因为其作者将其发布到了CRAN（`http://cran.r-project.org/web/packages/rmmseg4j/index.html`）。由于mmseg4j 之类的分词工具已经是之前好几代的技术，目前很少有人还在使用，所以对于R的用户造成了很大的困扰。很多用户之所以还在用这个包的原因是其能自定义词库。

我们首先使用基本的设置进行分词：

```
> require(rmmseg4j)
> mmseg4j(teststring1)
```

```
[1] "小 都 督促 织女 将 r 语言 学习 到底"
```

```
> mmseg4j(teststring2)
```

```
[1] "小 都是 统计 之都 的 昵称"
```

当前版本的rmmseg4j的默认词库中没有“促织”和“R语言”这两个词，因此我们将其添加到自定义词库（ %R_HOME%\library\rmmseg4j\userDic\words-rmmseg4j.dic），再进行分词：

```
"小 都督 促织 女将 r 语言 学习 到底"
```

可以发现分词全部乱了。进行10000次分词，测试时间：

```
> system.time(for(i in 1:10000) mmseg4j(teststring1))
```

```
   user  system elapsed
 171.91    8.83  180.24
```

花费了3分钟左右。

2.2 imdict-chinese-analyzer

imdict-chinese-analyzer是imdict智能词典的智能中文分词模块，作者高小平，算法基于隐马尔科夫模型（Hidden Markov Model, HMM），是中国科学院计算技术研究所的ictclas中文分词程序的重新实现（基于Java），可以直接为lucene搜索引擎提供中文分词支持。

ictclas应该是目前分词准确度和效率最好的工具，不过是商业软件。imdict-chinese-analyzer是它的一个开源的JAVA版本，但是不能支持词性识别和自定义词库。

本Rwordseg包作者的主页（`http://jliblog.com/app/rsegword`）上有一个集成了imdict-chinese-analyzer的RsegWord包，另外rmmseg4j 的作者还提供了一个rsmartcn，基于smartcn，这是imdict 集成到lucene之后的结果，效果和imdict-chinese-analyzer一样。

我们使用RsegWord进行测试[1]：

```
> require(RsegWord)
> segWord(teststring1)
```

```
[1] "小"   "都"   "督促" "织女" "将"   "r"    "语言" "学习" "到底"
```

```
> segWord(teststring2)
```

```
[1] "小"   "都"   "是"   "统计" "之"   "都"   "的"   "昵"   "称"
```

[1]该包已经停止维护，这里的例子只是为了说明新版和旧版的不同。

从teststring2的例子可以看到其并未将“都是”识别成一个词，是一个进步，但是词库中没有“昵称”这个词，也没办法添加。

我们再看其进行10000次分词的时间：

```
> system.time(for(i in 1:10000) segWord(teststring1))
```

```
   user  system elapsed
  25.35    1.04   22.20
```

时间缩短到了25秒。

2.3 ansj

前面已经介绍了Ansj 也是基于中科院的ictclas 中文分词算法，但是作者用JAVA重写了整个系统，并且全部开放源代码。

基础的分词功能和imdict 差别不大，不过基础词典有更新，包含了“昵称”一词。

```
> require(Rwordseg)
> segmentCN(teststring1)
```

```
[1] "小"   "都"   "督促" "织女" "将"   "r"    "语言" "学习" "到底"
```

```
> segmentCN(teststring2)
```

```
[1] "小"   "都"   "是"   "统计" "之"   "都"   "的"   "昵称"
```

我们同样将“促织”和“R语言”这两个词加入词库：

```
> insertWords(c("促织", "R语言"))
> segmentCN(teststring1)
```

```
[1] "小"    "都"    "督促"  "织女"  "将"    "r语言" "学习"  "到底"
```

可以发现分词器并未受到新词的干扰，仍然保持了对句子的正确理解，同时中英文混合的词“R语言”也能正确地分出来。

测试10000次分词的时间：

```
> system.time(for(i in 1:10000) segmentCN(teststring1))
```

```
   user  system elapsed
   7.75    0.02    6.57
```

可以发现性能有了进一步地提升，只花费8秒不到。

3 分词操作

3.1 默认分词

默认参数下输入需要分词的句子，GBK或者UTF-8的编码都可以，注意需要在R的控制台中显示正常。

```
> segmentCN("结合成分子时")
```

```
[1] "结合" "成"   "分子" "时"
```

如果输入参数为字符向量，则返回列表：

```
> segmentCN(c("说的确实在理", "一次性交多少钱"))
```

```
[[1]]
[1] "说"   "的"   "确实" "在"   "理"

[[2]]
[1] "一次性" "交"     "多少"   "钱"
```

参数nosymbol表示是否只输出汉字、英文和数字，默认为TRUE，否则将会输入标点符号等。

```
> segmentCN("想渊明、《停云》诗就，此时风味。")
```

```
[1] "想"   "渊明" "停"   "云"   "诗"   "就"   "此时" "风味"
```

```
> segmentCN("想渊明、《停云》诗就，此时风味。", nosymbol = FALSE)
```

```
 [1] "想"   "渊明" "、"   "《"   "停"   "云"   "》"   "诗"   "就"   "，"   "此时" "风味" "。
```

3.2 词性识别

参数nature可以设置是否输出词性，默认不输出，如果选择输出，那么返回的向量名为词性的标识：

```
> segmentCN("花一元钱买了一朵美丽的花", nature = TRUE)
```

```
    v     m     n     v    ul     m     a    uj     v
"花" "一元"  "钱"  "买"  "了" "一朵" "美丽"  "的"  "花"
```

不过目前的词性识别并不是真正意义上的智能词性识别，结果仅作为参考。

3.3 人名识别

默认的分词器会进行自动的人名识别，有时候会和自定义词库冲突，得不到正确的结果，可以将recognition参数设为FALSE使之不进行人名识别：

```
> insertWords(c("长焦", "微距"))
> segmentCN("长焦和微距")
```

```
[1] "长"     "焦和微" "距"
```

```
> segmentCN("长焦和微距", recognition = FALSE)
```

```
[1] "长焦" "和"   "微距"
```

4 词典管理

4.1 安装和卸载词典

该包支持安装新的词典，一次安装之后，每次重启R包事都会自动加载。目前支持普通格式的文本词典和Sogou的Secl格式细胞词典。

函数 installDict 用来安装新词典，参数 dictpath 表示词典的路径；参数 dictname 表示自定义的词典名称，需要输入英文单词，默认是“userDefine”；参数 dicttype 表示词典的类型，目前只支持“text”和“scel”，分别表示普通文本格式和Sogou细胞词典，默认为“text”，如果后缀为其他格式（比如.scel），则自动当作该格式导入。

普通文本格式的后缀名没有限制，ANSI和UTF-8的文件都可以支持，需要确保文件用编辑器打开时能正常显示中文。我们可以安装ansj项目中的自定义词库（https://github.com/ansjsun/ansj_seg/tree/master/library/userLibrary）。安装前：

```
> segmentCN("湖北大鼓真是不错啊")
```

```
[1] "湖北" "大鼓" "真"   "是"   "不错" "啊"
```

安装新的词典：

```
> installDict("E:/userLibrary.dic")
```

```
OK!
New dictionary was installed, please restart R to use it.
```

重启R则生效：

```
> require(Rwordseg)
> segmentCN("湖北大鼓真是不错啊")
```

```
[1] "湖北大鼓" "真"       "是"       "不错"     "啊"
```

在这里强烈推荐Sogou的细胞词库（http://pinyin.sogou.com/dict/），包含数目非常巨大的分类词典，而且可以找到最新的流行词。建议在进行具体研究之前找到相关的专业词典进行安装，比如要研究金庸的小说，可以下载词典“金庸武功招式.scel”。

```
> segmentCN("真武七截阵和天罡北斗阵哪个厉害")
```

```
[1] "真武" "七"   "截"   "阵"   "和"   "天罡" "北斗" "阵"   "哪个" "厉害"
```

可以先将.scel的词典导入到R从而查看该词典包含哪些内容：

```
> a1 <- importSogouScel("E:\\金庸武功招式.scel")
> attributes(a1)
```

```
$Type
[1] "读书"

$Description
[1] "金庸小说中的武功和招式名称"
```

安装新的词典：

```
> installDict("E:\\金庸武功招式.scel", dictname = "jinyong")
```

```
OK!
New dictionary was installed, please restart R to use it.
```

重启R则生效：

```
> require(Rwordseg)
> segmentCN("真武七截阵和天罡北斗阵哪个厉害")
```

```
[1] "真武七截阵" "和"         "天罡北斗阵" "哪个"       "厉害"
```

查看已安装的词典：

```
> listDict()
```

```
[1] "userDefine" "jinyong"
```

如果想恢复到原状，可以卸载某些词典，默认卸载所有：

```
> uninstallDict()
> listDict()
```

```
character(0)
```

重启R则生效：

```
> require(Rwordseg)
> segmentCN("湖北大鼓真是不错啊")
```

```
[1] "湖北" "大鼓" "真"   "是"   "不错" "啊"
```

4.2 自定义文本词典

如果仅仅只是用户自己添加词汇，没有必要做一个词典进行安装，可以使用自定义词典的功能。默认的词典目录为%R_HOME%\Rwordseg\dict。可以在其中添加任意后缀为.dic的文本，里面可以输入自定义的词，每一行一个词，回车换行。

如果改变了默认的路径，需要重新设置option:

```
> options(dic.dir = "c:/new/path")
```

修改后每次重启时都会导入用户自定义词典。如果想要立即生效，可以在修改后运行：

```
> loadUserDict()
```

4.3 手动添加和删除词汇

如果仅仅只是在内存中临时添加或者删除词汇，可以使用insertWords和removeWords：

```
> segmentCN("画角声断谯门")
```

```
[1] "画角" "声"   "断"   "谯"   "门"
```

```
> insertWords("谯门")
> segmentCN("画角声断谯门")
```

```
[1] "画角" "声"   "断"   "谯门"
```

```
> removeWords(c("画角", "谯门"))
> segmentCN("画角声断谯门")
```

```
[1] "画" "角" "声" "断" "谯" "门"
```